# Towards Quantifying Commonsense Reasoning with Mechanistic Insights

**Abhinav Joshi** **Areeb Ahmad** **Divyaksh Shukla** **Ashutosh Modi**
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur (IIT Kanpur)
{ajoshi,areeb,divyaksh,ashutoshm}@cse.iitk.ac.in

## Abstract

Commonsense reasoning deals with the implicit knowledge that is well understood by humans and typically acquired via interactions with the world. In recent times, commonsense reasoning and understanding of various LLMs have been evaluated using text-based tasks. In this work, we argue that a proxy of this understanding can be maintained as a graphical structure that can further help to perform a rigorous evaluation of commonsense reasoning abilities about various real-world activities. We create an annotation scheme for capturing this implicit knowledge in the form of a graphical structure for 37 daily human activities. We find that the created resource can be used to frame an enormous number of commonsense queries ($\sim 10^{17}$), facilitating rigorous evaluation of commonsense reasoning in LLMs. Moreover, recently, the remarkable performance of LLMs has raised questions about whether these models are truly capable of reasoning in the wild and, in general, how reasoning occurs inside these models. In this resource paper, we bridge this gap by proposing design mechanisms that facilitate research in a similar direction. Our findings suggest that the reasoning components are localized in LLMs that play a prominent role in decision-making when prompted with a commonsense query.

## 1 Introduction

The growth of Large Language Models (LLMs) performing well on a wide variety of commonsense reasoning benchmarks (West et al., 2023; Bosselut et al., 2019; Hwang et al., 2021; Park et al., 2020) raises the question of whether LLMs are truly capable of reasoning in a more practical setting of real-world daily activities that involve commonsense. Though in the past, a wide range of benchmarks/datasets (information sources) have been proposed, building a benchmark with exhaustive and rigorous analysis has always remained

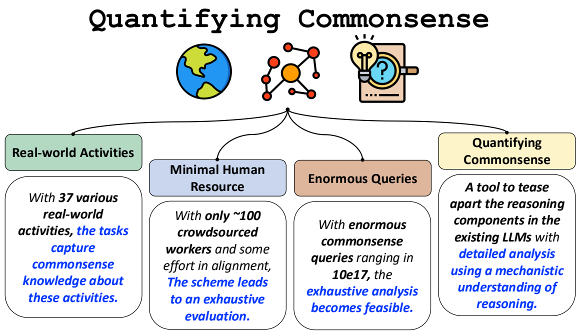


Figure 1: Quantifying commonsense reasoning in Large Langauge Models (LLMs).

a challenge. To quantify the commonsense reasoning abilities of LLMs in an exhaustive manner, one would require a few primary features about an information resource 1) the information source should consider real-world tasks, well understood by humans (capturing commonsense) 2) the information resource should be exhaustive, containing all possible ways of performing a task, and, 3) the information resource should support creating reasoning questions, that help in understanding of reasoning mechanisms of models via marginalization with multiple samples. We found that *"Scripts"* (Schank, 1975; Schank and Abelson, 1975) help create a tangible framework that satisfies all these requirements. Scripts are defined as a sequence of events describing a prototypical activity, such as 'going to a restaurant,' 'baking a cake,' etc., capturing commonsense knowledge about the world (Schank and Abelson, 1975; Modi et al., 2016; Wanzare et al., 2016; Ostermann et al., 2018; Modi, 2016, 2017; Modi et al., 2017; Modi and Titov, 2014). Since all the real-world tasks are generic, writing about steps/events while performing the activity can be done in an enormous number of different ways. Additionally, these activities are easy to reason about, and previous works (Modi and Titov, 2013, 2014; Modi et al., 2017) have used



*Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 9633–9660
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

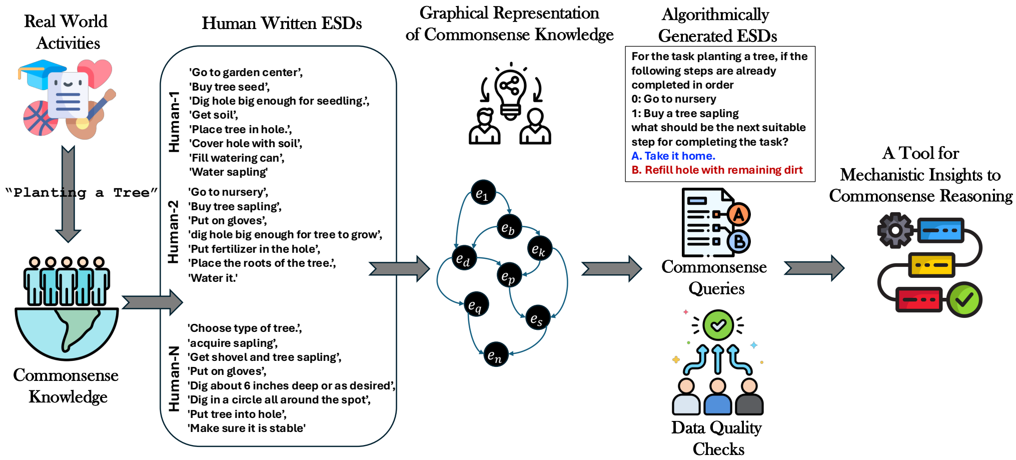


Figure 2: The figure provides an overview of the proposed resource. Real-world activities (well understood by humans) are considered to capture commonsense knowledge about these activities via human crowdsource workers. These ESDs are used to create a graphical representation of these activities and the underlying commonsense knowledge. The graphical representations help create enormous commonsense queries ($\sim 10^{17}$ queries per activity). The created resource of commonsense queries is reverified via data quality checks from humans. The overall flexibility attained using the graphical representations helps tease apart the reasoning mechanisms of LLMs, creating a tool for mechanistic insights into commonsense reasoning.

them to create commonsense reasoning queries, assessing the quality of acquired commonsense knowledge. Moreover, they inherently provide a structure that helps facilitate marginalization across different variations, opening new directions for localizing information (Meng et al., 2023; McGrath et al., 2023; Wang et al., 2022; Goldowsky-Dill et al., 2023) contained in the decision-making process of commonsense reasoning.

In this work, we propose a generic scheme for rigorously evaluating commonsense knowledge and understanding of LLMs via commonsense reasoning questions. For a framework devised to validate the commonsense understanding of implicit commonsense knowledge, it becomes imperative to consider the dataset directly coming from humans (i.e. written and annotated by humans with minimal synthetic intervention). Hence, for our analysis, we consider a crowdsourced commonsense resource about daily human activities called as DeScript (Wanzare et al., 2016). We create a directed graph from the DeScript corpus, which is subsequently used (via an algorithm) to generate commonsense reasoning questions about various activities. LLMs are then evaluated for commonsense reasoning via these questions. Further, we investigate where does commonsense reside in the pretrained autoregressive transformer-based models. In particular, we use activation path patching to localize the decision-making for commonsense reasoning in these models. We find that the proposed framework provides promising flexibility for such analysis and will help facilitate future research in Mechanistic Interpretability for commonsense reasoning. We make the following contributions:

- We provide resources for creating directed commonsense knowledge graph for 37 scenarios (daily human activities). These graphical representations of human activities are suitable to act as a proxy for comprehending the underlying commonsense knowledge about these activities. Fig. 1 shows the key features of the framework.
- We propose a generic scheme (based on graphs) to create prompts that help validate the commonsense reasoning and language understanding.
- Via experimentation with 6 open-weight models `gpt-neo-1.3B` (Black et al., 2021), `gpt-j-6B` (Wang and Komatsuzaki, 2021), `phi-2` (Javaheripi et al., 2023), `Llama2-7b` (Touvron et al., 2023), `Mistral-7B` (Jiang et al., 2023), and `Llama-3-8B` (Grattafiori et al., 2024), we highlight trends and gaps in commonsense knowledge, understanding, and reasoning abilities of these models.
- As a use-case for the proposed dataset, with the aim of understanding the reasoning process, we propose design mechanisms to tease apart the decision-making happening inside pretrained LLMs. The high flexibility of the proposed framework helps to provide a more decisive finding about commonsense reasoning happening inside these models.

- We perform localization experiments over `phi-2` (being both computationally moderate with better performance) and investigate the commonsense knowledge reasoning in detail. We release the dataset/code via GitHub: https://github.com/Exploration-Lab/CoReMech.

## 2 Methodology

In this section, we provide details about the scheme created for a rigorous/exhaustive analysis of commonsense reasoning abilities related to daily real-world activities and how an enormous number of commonsense queries (ranging $\sim 10^{17}$ on average per scenario) can be created to evaluate the quality of commonsense understanding in LLMs. (Figure 2 provides an overview of the proposed scheme)

**Dataset:** As outlined earlier, we use crowdsourced resource DeScript (Wanzare et al., 2016), which provides a telegrammic-style version of script event sequences (referred to as Event Sequence Descriptions (ESDs)) for various stereotypical human activities. DeScript provides a list of 40 stereotypical human activities (each referred to as a scenario or activity) along with $\sim$ 100 ESDs provided by crowd-sourced workers for each of the 40 scenarios. DeScript also annotates 10 scenarios by grouping similar events (telegrammic steps). For example, in a scenario like "`Washing Dishes,`", the events like "`dry utensils`" and "`clean utensils with a clean, dry cloth`" are grouped. In this work, we extend the annotations and provide the alignments for the remaining 30 scenarios, leading to a rich resource of 37 daily activity scenarios (3 scenarios are discarded as these were found to be too noisy). We create a directed graph with the help of aligned sequences (coming from annotations), consolidating information supplied by $\sim$ 100 crowd workers into a single graph. Using the graph, we devise a scheme to generate commonsense reasoning questions about these activities. The complete list of the considered scenarios is presented in Table 1.

**Annotations and Event Alignments:** Though there can be multiple ways (various ESDs) of describing script for a scenario, there exists an alignment among events in multiple descriptions. The alignments assign generic groups to an event. For example, events like "`go inside the car,`" "`get into your car,`" "`enter the car,`" etc., are assigned a group like "`get-into-car.`" DeScript provides these alignments between the events for only 10 scenarios. We extend these alignment annotations and perform the annotations for all 40 scenarios. A group of 3 annotators (graduate students) performed all the annotations as a part of a course research project. Annotators were asked to make generic clusters to perform the specific task and assign each event to these clusters. It took around 4-12 hrs (spanning across a month) for an annotator to complete annotations and alignments for a scenario. The varied amount of time highlights the task complexity and variety in descriptions. Further, we manually inspected the alignments and found the quality of the 3 scenarios to be too noisy, and we discarded these. Hence, only 37 were used in the final analysis.

| Scenario/Activity | Deg. | Total possible ESDs |
|---|---|---|
| `baking a cake` | 3.6 | $4.0e+26$ |
| `borrowing book from Library` | 3.7 | $3.1e+19$ |
| `changing batteries in alarm clock` | 5.8 | $8.1e+19$ |
| `checking in an airport` | 8.6 | $7.7e+23$ |
| `cleaning up a flat` | 7.4 | $1.1e+20$ |
| `cooking pasta` | 5.4 | $1.1e+22$ |
| `doing laundary` | 9.5 | $5.0e+38$ |
| `eating in a fast food restaurant` | 6.7 | $6.9e+27$ |
| `flying in an airplane` | 3.6 | $2.6e+30$ |
| `fueling a car` | 8.2 | $4.6e+29$ |
| `getting a haircut` | 3.7 | $4.0e+28$ |
| `going grocery shopping` | 3.7 | $2.3e+26$ |
| `going on a Train` | 3.7 | $3.1e+21$ |
| `going to the dentist` | 6.6 | $7.8e+23$ |
| `going to the swimming pool` | 7.2 | $1.5e+16$ |
| `going to the theatre` | 6.3 | $8.1e+16$ |
| `going to the sauna` | 7.3 | $1.3e+22$ |
| `going bowling` | 9.5 | $1.8e+37$ |
| `having a barbeque` | 6.8 | $6.5e+20$ |
| `ironing Laundary` | 7.8 | $2.1e+36$ |
| `making scrambled Eggs` | 7.9 | $4.0e+30$ |
| `making a bonfire` | 8.0 | $3.5e+20$ |
| `making a coffee` | 8.0 | $9.8e+21$ |
| `paying with a credit card` | 7.8 | $2.4e+21$ |
| `planting a Tree` | 3.7 | $1.6e+16$ |
| `playing Tennis` | 6.7 | $1.1e+18$ |
| `renovating a room` | 8.3 | $3.1e+31$ |
| `repairing flat bicycle Tire` | 3.4 | $8.4e+18$ |
| `riding on a bus` | 3.8 | $1.0e+17$ |
| `sewing a button` | 7.5 | $7.7e+28$ |
| `taking a bath` | 3.7 | $3.1e+27$ |
| `taking a shower` | 7.6 | $2.2e+30$ |
| `taking a driving lesson` | 7.9 | $3.2e+15$ |
| `taking a child to bed` | 3.7 | $4.4e+15$ |
| `washing ones hair` | 7.4 | $8.8e+34$ |
| `washing dishes` | 7.6 | $7.3e+27$ |

Table 1: The table provides details of the generated graphs for 37 scenarios.

**Remark:** Unlike classification tasks, clustering doesn't typically have predefined categories. This makes it harder to establish a common framework for agreement between the annotators. Moreover, clustering comparison is challenging, and though there are metrics for comparing clusterings (e.g., Rand Index (Rand, 1971), Adjusted Mutual Information (Vinh et al., 2009), Fowlkes–Mallows index (Fowlkes and Mallows, 1983), etc.), a robust widely accepted metric for annotator agreement in clustering tasks is not readily available. It is to be noted that we create the dataset by sampling trajectories from the created DAG, which shows a way of performing the entire activity. Hence, we used the same to assess the quality of the annotations and made suitable changes by manual inspection. Note that although the clustering annotations may vary (in terms of granularity), the final task (defined in the later section) is only dependent on the trajectory sequence, making it suitable for the generated commonsense queries.

**Graphical Representation:** Taking inspiration from (Joshi et al., 2023), we create a graph structure (also referred to as Scenario Compact Graph or Compact Graph for short) from event alignments. In the graphical representation, each cluster (group) is a node in the graph. For connecting the nodes with directed edges, we use the original description sequence provided in the ESDs. In particular, a directed edge is drawn from node $p$ to $q$ if there is at least one action (telegram-style step description) in node $p$ that directly precedes an event in node $q$. This simple strategy leads to a rich graph structure of scenarios that resembles the human understanding of these tasks. Fig. 7 shows an example of such a graph. These directed acyclic graphs (DAGs) provide a medium for generating enormous trajectories (refer to Table 1), that are coming directly from human annotations (alignment annotation as well as the ESDs written by crowd-soured workers), providing us a proxy to represent the understanding of daily activities. We provide more details about graphical representations and computing the total number of ESDs in the App. A.

**Trajectory Entropy:** To quantify the complexity across various scenarios and compare the created graphical representations in detail, we also define Trajectory Entropy $\mathcal{H}_t$ (details in App. D). Fig. 10 provides a comparison of various scenarios in terms of number of paths and the defined Trajectory Entropy $\mathcal{H}_t$.

**Reasoning Question Creation:** To test LLMs for commonsense knowledge understanding, we would like to generate commonsense reasoning questions related to the obtained activities. We generate questions via compact graphs. Each path in the compact graph denotes a suitable set of steps (events) for accomplishing a task. Using the graph, we sample multiple trajectories for finishing the task $t_1, t_2, \ldots, t_n \in \tau$. Each of these trajectories contains multiple events of ESDs, $e_1, e_2, \ldots e_{m_{t_i}}$. Note, since different trajectories may require different numbers of steps, $m_{t_i}$ (referred to as $m$ when it is clear from the context) is a random variable here, which depends on the selected trajectory $t_i$. Given a trajectory, we further use a subpart of the trajectory by taking a split at a step $n \in \{1, m\}$ and use steps $e_1, e_2, \ldots e_{n-1}$ as a part of a commonsense reasoning question and $e_n$ as the correct choice for the question. Using the obtained samples, we use a template prompt to generate a commonsense reasoning question. App. Fig. 8 shows a template prompt.

**Data Quality Check:** A noteworthy point about the created dataset is that although it is generated using an algorithmic procedure, the core knowledge still comes from humans. The algorithmic generation provides an added advantage of exhaustiveness with a meager human annotation cost, making the generated distribution of commonsense queries less likely to be previously seen by the pretrained LLMs. We additionally perform some manual inspection to improve the dataset quality (details in App B). Lastly, we conducted a sanity check, where we took a sample of 1k commonsense queries for 5 of 37 scenarios and asked 5 human annotators to know how well humans perform on the created task. We recorded an average accuracy of 95% with 92% and 98% being the minimum and maximum, respectively (more details in Table 2), validating the commonsense captured by the created queries. Interestingly, we ran an evaluation over the same set of 1k queries using one of the proprietary-LLM (`‘claude-3.5-sonnet-20240620’`) and observed a success rate of 94.30%, which is very close to human performance.

To this end, the proposed scheme can create an enormous number of commonsense queries ($\sim 10^{17}$ for a single activity), facilitating a rigorous/exhaustive evaluation of commonsense knowledge about these activities.

**Remark on Terminology:** We use the word "exhaustive" specifically in reference to the procedural knowledge captured in the DeScript corpus, which denotes comprehensive coverage of event orderings and dependencies for the human-authored activities in our framework, not universal commonsense knowledge which varies culturally as well as contextually. The proposed scheme enables testing over enormous trajectories per activity, exhausting the solution space defined by the original crowd workers' procedural annotations, making it a suit-

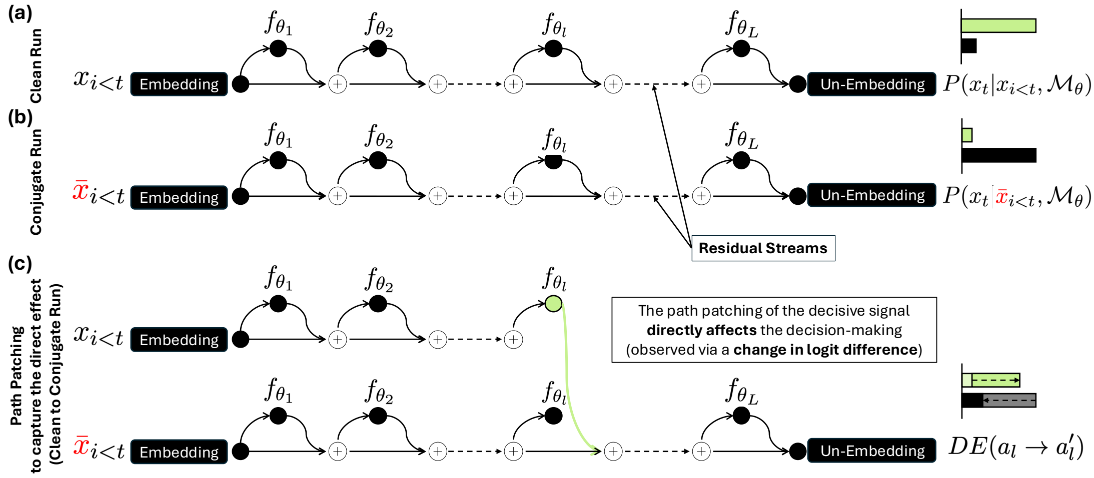


Figure 3: The figures highlight the computation of direct effect via path patching. **(a)** A run with the clean prompt ($x_{i<t}$) is passed through the model, saving all the intermediate states. (b) A model pass is again done using a conjugate prompt ($\bar{x}_{i<t}$) that flips the expected behavior of the model from green option to black option. (c) A run for computing the direct effect is done, where a path patching takes place for $f_{\theta_l}$, i.e., the green signal is patched to the conjugate run. The change in logit values helps localize the decision-making component that plays a vital role in the model selecting green as the correct choice.

able proxy for capturing the underlying commonsense knowledge in these activities.

## 3 A Tool for Mechanistic Insights

In recent times, pretrained transformer-based networks have shown remarkable performance in a wide range of tasks (Devlin et al., 2019; Radford et al., 2019; Brown et al., 2020), including some of the popular commonsense reasoning tasks (Zellers et al., 2019a; Zhao et al., 2023). However, the understanding of decision-making happening inside these large models remains limited. With the help of the proposed dataset generation scheme, we would like to investigate how a commonsense reasoning query is answered by these large decoder-only autoregressive transformer-based language models autoregressive transformer models.

Though there have been some works localizing the information in these models (Wang et al., 2022; Meng et al., 2023; Goldowsky-Dill et al., 2023), tools to tease apart the decision-making happening inside these models remain limited. We investigate if the decision-making in these commonsense reasoning queries can be localized.
A prompt acting as an input to a Language Model (LM) comprises information related to the query that helps determine the expected answer. In our setup, we focus on the multiple-choice question answering (MCQA) prompt, which consists of two critical components, **1) Incomplete Task Trajectory** ($traj.$): which includes the sequence of states or steps, capturing the partial progression toward completing the task. **2) A Choice Set** (**A.** $o_{correct}$; **B.** $o_{wrong}$)) consisting of two options from which the LM must select the correct answer and generate as output either **A** or **B**. Note that the **A.** and **B.** are for representation, and in the actual run, the correct/wrong options are shuffled to marginalize the effect of models choosing a specific option.

The decision taken by the LM (($\mathcal{M}_\theta$), where $\theta$ represents the model parameters) depends on these two critical components. Additionally, the predictions also depend on the way in which the query is framed, i.e. the prompt template ($x_\epsilon$) used to frame the queries. The predicted probability/logit value of the next token can be written as

$$
\begin{aligned}
P(x_t|x_{i<t}, \mathcal{M}_\theta) &= P(x_t|x_{traj.}, x_{options}, x_\epsilon, \mathcal{M}_\theta) \\
x_{traj.} &\leftarrow \{s_1, s_2, \ldots, s_n\} \\
x_{options} &\leftarrow \{\mathbf{A.}\ o_{correct}, \mathbf{B.}\ o_{wrong}\} \\
x_\epsilon &\in \text{set of prompt templates} \\
\mathcal{M}_\theta &= \{f_{\theta_1}, f_{\theta_2}, \ldots f_{\theta_L}\}
\end{aligned}
$$

In the transformer-based language model, the input prompt ($x_{i<t}$) is passed through a sequence of transformer blocks/layers ($f_{\theta_1}, f_{\theta_2}, \ldots f_{\theta_L}$), providing a distribution of logits over the vocabulary for the next tokens, we only consider the predicted distribution of the last token ($x_t$), i.e., the token responsible for answering the reasoning query (using logits corresponding to tokens '_A' and '_B', see

Fig. 8 for reference).

$$\mathcal{M}_\theta(x_{i<t}) = f_{\theta_L}(1 + f_{\theta_{L-1}}(\ldots(1 + f_{\theta_1}(x_{i<t})))$$

These sequences of operations play a crucial role in modifying the residual stream (the $1+$ denotes the update in the residual stream throughout the transformer blocks), leading to the final predicted token $x_t$. Fig. 3 (a) highlights the signal passing through the residual stream where transformer blocks are present in parallel. Note, in some of the transformer implementation designs, there are two points in a single transformer block where the computational blocks read/write back from/to the residual stream (self-attention and MLP); we skip the mid-skip connection in the equations above for brevity.

**Direct Effect:** To measure the effect of the transformer's $l_{th}$ layer over the predicted decision, we make use of the direct effect, we follow Chattopadhyay et al. (2019); Meng et al. (2023); McGrath et al. (2023); Heimersheim and Nanda (2024) assuming the transformer-based architectures as structural causal models (SCMs) (Pearl et al., 2016). The direct effect of intervening over the activations $A_l = a_l \rightarrow A_l = a'_l$ is computed as

$$\begin{aligned} DE(a_l \rightarrow a'_l) = \\ & P(x_t \mid do(A_l = a'_l, A_{\neq l} = a_{\neq l}(x_{i<t}))) \\ & - P(x_t \mid do(A_l = a_l(x_{i<t}))) \end{aligned}$$

where $do(.)$ denotes the do operator (Pearl, 2012) showing the intervention on $A_l$, i.e., estimating the effect of intervening at the $l_{th}$ layer's activation $A_l$ and setting the value to $a'_l$, keeping all the other activations intact $A_{\neq l} = a_{\neq l}(x_{i<t})$ to the value that they would have when passing $x_{i<t}$ as input prompt. The second term helps capture the effect, representing the model output, i.e. $P(x_t \mid do(A_l = a_l(x_{i<t}))) = P(x_t|x_{i<t})$. This way of computing the intervention via replacing activations is also known as *path patching* (Wang et al., 2022; Goldowsky-Dill et al., 2023) (also see Fig. 3). Essentially, the direct effect measures how much changing the activation would affect the output logits if all other units were kept constant, i.e., in the setup of a language model, only units that are connected via the residual path to the output can have a direct effect.

**Intervention with Corrupted run:** A crucial aspect of capturing the direct effect is the choice of clean and corrupted runs. A clean run denotes the expected behavior. In contrast, a corrupted run signifies changes in the inputs that disrupt/deviate the expected behavior. To localize the decision-making happening in the network parameters, we take a corrupted run and intervene over the activations via representations coming from the clean run. We further observe which interventions restore the expected behavior, highlighting the components that play a vital role in commonsense reasoning. Another common, widely used strategy is to patch the clean run over the corrupted run, where a Gaussian Noise is added to the same clean input (also known as Causal Tracing (Meng et al., 2023)). Some of the previous works (Heimersheim and Nanda, 2024) highlight the significance of constructing a corrupted run via similar prompts (or counterfactual prompts), making them more decisive in comparison to other methods. The flexibility in the proposed framework of commonsense queries coming from a DAG opens up a wide scope for constructing such queries.

**Conjugate Prompts:** To be more decisive in the decision-making via path patching. We define a new way of constructing the corrupted run prompts. We call these Conjugate Prompts. For any query prompt ($x_{i<t}$), we can construct a conjugate query prompt by replacing the trajectory tokens with trajectory where the wrong option becomes the correct choice and vice versa, keeping the set of choices in the prompt intact. App. Fig. 9 provides a pair of conjugate prompt templates. This strategy helps capture the specific dependency on the trajectory, and after sampling multiple such trajectories, one could be more decisive about the localization of decision-making in the clean trajectory. Note that the constructed query consists of multiple segments

$$\begin{gathered} P(x_t|x_{i<t}, \mathcal{M}_\theta) = P(x_t|x_{traj.}, x_{options}, x_\epsilon, \mathcal{M}_\theta) \\ x_{traj.} \leftarrow \{s_1, s_2, \ldots, s_n\} \\ x_{options} \leftarrow \{\textbf{A.}\ o_{correct}, \textbf{B.}\ o_{wrong}\} \\ x_\epsilon \in \text{set of prompt templates} \end{gathered}$$

and the $o_{correct}$ is the $s_{n+1}$ whereas the $o_{wrong}$ comes from a randomly sampled node (far from the current node) of the compact graph . For the construction of a corrupted prompt that provides a decisive distinction, one would need a prompt that flips the answer. We create such prompts by taking the $o_{wrong}$ and sample a conjugate trajectory that starts at the start node and ends at the wrong node ($o_{conjugate} \leftarrow o_{wrong}$). We further construct the conjugate prompt ($\bar{x}_{i<t}$) by replacing the $x_{traj.}$ with $x_{\bar{traj}.}$.

$$\bar{x}_{i<t} = x_{\bar{traj}.} + x_{options} + x_{\epsilon}$$
$$P(x_t|\bar{x}_{i<t}, \mathcal{M}_\theta) = P(x_t|x_{\bar{traj}.}, x_{options}, x_\epsilon, \mathcal{M}_\theta)$$

Note that the original clean run still remains the same with the same set of options present in the prompt.

$$x_{i<t} = x_{traj.} + x_{options} + x_{\epsilon}$$

This minimal control helps flip the decision of a language model as for the conjugate prompt, the conjugate (wrong for clean) becomes the right choice. The direct effect of path patching on the $l_{th}$ layer, from clean run to conjugate run will be

$$\begin{aligned} DE(a_l \to a_l') = \\ & P(x_t|do(A_l = a'_l, A_{\neq l} = a_{\neq l}(\bar{x}_{i<t})) \\ & - P(x_t|\bar{x}_{i<t}) \end{aligned}$$

where $a_l'$ comes from the clean run, and the remaining activations are set from the conjugate run ($A_{\neq l} = a_{\neq l}(\bar{x}_{i<t})$). Fig. 3 highlights the overall mechanism in detail, where the clean run predicts the green option being correct, whereas, for the corrupted, the model predicts the option highlighted using a black bar. Further, intervening in the signals via path patching from the clean run to the corrupted run shows the expected clean behavior (green being higher) when a decisive signal is patched from the clean run. To capture the decision-making process, we monitor the deviations in the logits of the predicted options (i.e., the logits corresponding to '_A' and '_B' tokens). Given the flexibility of sampling multiple such prompts, a more conclusive result about the localization of decision-making can be made.

## 4 Experimental Setup: Evaluating LLMs

We experiment with multiple (6) open-weight autoregressive models that are widely used by the community. We specifically make use of open-weight models to consider for easier replication of results and empirical transparency. Note that the primary aim of these experiments is not to benchmark the state-of-the-art models but to demonstrate the utility of the created resource for rigorous evaluation, and to enable interpretability studies in regard to commonsense understanding in LMs.

**MCQA based Evaluation of Open-Weight Models:** For a prompt-based evaluation scheme,

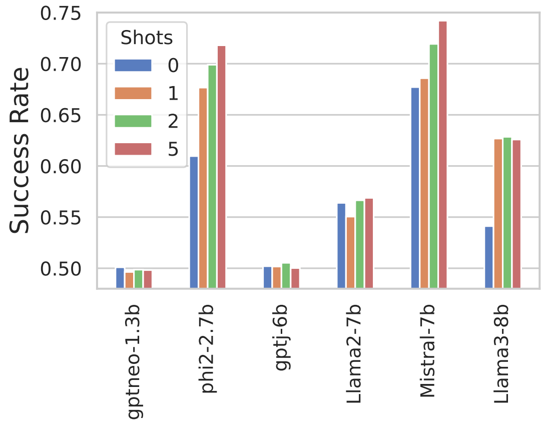


Figure 4: Success rates of different models compared across the number of shots of in-context examples.

we frame the prompt as a multi-choice question answering (MCQA) objective (Robinson and Wingate, 2023). The prompt is intentionally structured so that the LLM is intended to predict a single-choice token (Such as 'A,' 'B,' etc.). Robinson and Wingate (2023) highlight the advantages of MCQA-based evaluation over cloze evaluation (Brown et al., 2020) (where the LLMs are expected to generate the entire answer in a cloze test), leading to a significant boost in performance over various tasks, including commonsense-based tasks. Fig. 8 shows prompt templates with a qualitative example of the framed commonsense reasoning query. Additionally, to validate the effectiveness of these open-weight models over the created resource, we also include additional experiments: **1) In-Context Learning**, **2) Fine-tuning** over the generated dataset, and **3)** Investigate the **generalization** between similar scenarios in detail. We provide details of these extended experimental setups in Appendix E.

## 5 Results and Empirical Findings

In this section, we provide an in-depth insight into the model's behavior over different aspects of the created commonsense queries.

**Overall Performance:** Table 4 shows success rates (i.e., total percentage of commonsense queries, where the LLM generates the expected correct option) for different models on a zero-shot task over all 37 scenarios. `Mistral-7b` shows the best performance, outperforming the other models comprehensively in the majority of the scenarios. Surprisingly, we observe that `phi-2`, which is a low-parameter model, slightly outperforms it in some

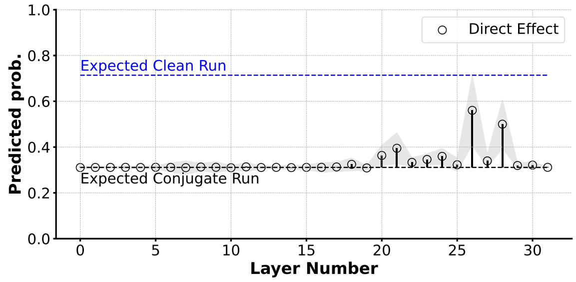


Figure 5: The figure shows the direct effect of path patching from the clean run to the conjugate run ('*going bowling*'), leading to deviations starting at layer 20 and increased signal strength at layer 26, highlighting the role of particular layers in commonsense reasoning.

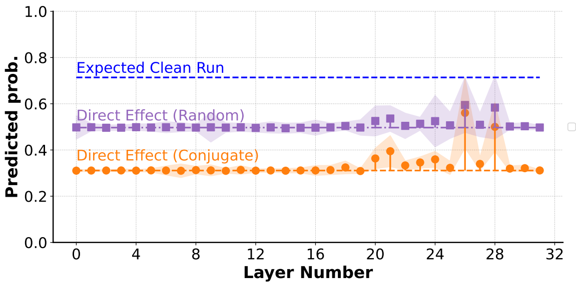


Figure 6: The figure shows the comparison of the direct effect of path patching from the clean to the random run and clean to conjugate run ('*going bowling*'). The peaks/deviations for the clean→random run are less decisive than the clean→conjugate run patching.

scenarios. The results are contrary to what is expected since the performance does not scale up with the number of parameters of the model, i.e., `phi-2` outperforms `gpt-j-6B, Llama-2, and Llama-3`, and despite having fewer parameters. A similar thing can be observed in the case of `Mistral-7B` better performing than `Llama-3-8B`. Fig. 4 highlights the success rates of each model across all the scenarios when prompted with zero-shot or few-shot examples of selecting the next steps in a task. We observe that `phi2-2.7b` and `Mistral-7b` show the best performance, and their performance rises as we increase the number of in-context examples. Additionally, we perform a detailed analysis of the obtained results to better understand the behavior of these models on the created commonsense queries across 37 scenarios. Due to space limitations, we discuss the remaining analysis in the App. G.

Overall, we find that pre-trained `phi2` (not fine-tuned on the specific tasks) with 2.7b parameters to be providing a decent performance performance with an average of 60.67% when compared to other models with a lower number of parameters. We choose `phi2-2.7b` to perform the localization in the decision-making experiments.

## 6 Localizing Commonsense Reasoning

To localize the components that play a primary role in the decision-making inside these models, we use the conjugate prompts (as previously explained). For these experiments, we consider a subset of the dataset (200 queries) for which we construct the conjugate prompts. Considering the actual performance of the phi model to be around 60%, we only select commonsense queries where the model predicts the correct choice. Fig. 5 shows the direct effect of path patching from the clean run to the conjugate run (for the scenario ('*going bowling*')) for different transformer layers. For the initial 20 layers (layer 0 to layer 19), we observe a minimal deviation in the predicted choice from the expected conjugate run. In contrast, after 20 layers, we start observing the shift of the predicted probabilities toward the Expected Clean Run, pointing toward the patched signal being responsible for decision-making. We hypothesize layer 20 to be the primary initiator of the decision-making, and the following layers increase the strength of (or help reinforce) the decision (layer 26 to show the maximum deviation). We perform a detailed set of these experiments over all the 37 scenarios present in the proposed framework. Interestingly, we find that these deviations are consistent across different scenarios (see App. Fig. 19), and there seems to exist a few specific modules that show a peak in the direct effect, pointing towards the localization of the decision-making component present in these large autoregressive models.

We also observe that there is an increase in peak detection when computing the direct effect from conjugate prompts (Fig. 5) when compared to a corrupted run created using a prompt with random tokens (Fig. 12). (also see Fig. 6 for comparison). This highlights the effectiveness of the proposed conjugate prompts, making the direct effect peaks more decisive for localizing the decision-making.

## 7 Related Works

The proposed scheme primarily targets a special case of commonsense reasoning. In the past, a large body of research works have investigated commonsense knowledge. Our work intersects with three broad research areas: 1) Commonsense Knowledge Resources, 2) Script-based Procedural Reasoning, and 3) Mechanistic Interpretability.

**Commonsense Knowledge Resources:** Some of the recent works to model commonsense reasoning include knowledge graphs like ATOMIC (Hwang et al., 2021), which captures social and physical inferences, and transformer-based generators like COMET (Bosselut et al., 2019) and the follow up works (West et al., 2023; Park et al., 2020; Choi, 2022; Rashkin et al., 2018). While these resources enable broad reasoning, they lack a granular procedural structure. On the other hand, benchmarks such as SWAG (Zellers et al., 2018), HellaSwag (Zellers et al., 2019b), and COIN (Ostermann et al., 2019) evaluate isolated inferences but ignore to test multi-step reasoning in procedural text. Some other works include (Qin et al., 2019; Huang et al., 2019; Bhagavatula et al., 2020; Qin et al., 2021; Talmor et al., 2021; Zellers et al., 2021; Zhao et al., 2024) Recently proposed methods show a good performance on these tasks (Lourie et al., 2021; Zhou et al., 2023), yet their performance remains limited to a small evaluation set, making quantification challenging. It is often difficult to quantify if the performance reflects surface pattern matching or structured understanding (Wang et al., 2024). Unlike these works, our scheme models activities as directed graphs, enabling evaluation through sampling enormous trajectories per activity.

**Script-Based Commonsense Reasoning:** Scripts have been an active area of research for the last four decades. Scripts provide a framework to formalize procedural knowledge as event sequences (Schank, 1975; Schank and Abelson, 1975), with corpora like InScript (Modi et al., 2016), DeScript (Wanzare et al., 2016), and McScript (Ostermann et al., 2018), capturing commonsense knowledge via crowdsourcing. Several computational models have developed to model script knowledge, *inter-alia*, (Regneri et al., 2010; Fremann et al., 2014; Modi, 2016; Modi and Titov, 2014; Rudinger et al., 2015; Jans et al., 2012; Pichotta and Mooney, 2016; Modi et al., 2017; Sancheti and Rudinger, 2022; Tandon et al., 2019; Madaan et al., 2021; Sakaguchi et al., 2021; Saha et al., 2021; Li et al., 2023; Creswell et al., 2023; Gandhi et al., 2023; Onoe et al., 2023; Poesia et al., 2023; Joshi et al., 2024). However, evaluations remain limited to small test sets and are often limited in capturing real-world variation. In this work, we expand this paradigm by converting scripts into directed graphs that encode valid event orderings per activity, supporting systematic stress-testing through marginalization over enormous trajectories.

**Mechanistically Interpretable Localization:** In recent years, a wide range of approaches have been proposed in the context of factual recall (Meng et al., 2023; Heimersheim and Nanda, 2024; Wang et al., 2022; Gordon et al., 2012), where the recalling circuit for a particular fact is found via circuit-level attribution in language models. A representative work widely used across these methods is the counteract dataset (Meng et al., 2023), which provides flexibility in choosing the counterfactual statement. Specifically, the dataset consists of a series of prompts and combines a tuple (subject, relation, object), and the object is replaced by a counterfactual object, making sense in the context. This helps tease apart the factual recalling mechanism by producing prompts whose completion requires specific factual knowledge about a subject and a relation. However, most of the prior art focuses on attribute recall rather than procedural reasoning. In this work, we extend it for commonsense reasoning happening inside these large autoregressive models by providing resources that facilitate marginalization using multiple samples.

## 8 Conclusion

In this work, we study to quantify commonsense knowledge acquired in LLMs by performing a rigorous evaluation over real-world activities well understood by humans. We provide alignment resources for 37 daily human activities, which can generate an enormous number of choice-based questions for validating the commonsense reasoning in LLMs. With a detailed analysis of 6 open-weight models, we find commonsense reasoning challenging for LLMs. To add an extra layer of understanding of the performance, we dive deeper into the relationships with different properties of the scenarios and report the findings. Further, we provide ways in which the decision-making about commonsense reasoning happening inside these models could be localized and understood. Our analysis using the Phi-2 model points out a few localized layers that play a crucial role in predicting the expected reasonable answer. We hope that this work opens up new ways of understanding the commonsense reasoning happening inside these models, by not only grasping the representations learned by these models but also by making a comparison with the compact graph representation of the commonsense knowledge about these daily real-world activities.

## Limitations

The major limitation of this work is the low number of stereotypical human activities (37 in number) used to validate the commonsense understanding aspect of LLMs. Though the validation space generated by the graph representation is enormous, the provided resource can only validate the commonsense understanding aspect in models for a limited set of these 37 scenarios, which may not be the true representative of the generalized understanding activities in the wild.

Though the framework supports the flexibility of choosing a set of question prompt templates, for our experiments, given the computation cost, we find a single prompt template that shows a nominal performance and use the same for all the analyses. In the future, it would be good to marginalize the results by using multiple prompt templates.

For finding the decision-making components in the large autoregressive language models, though we provide a rich resource that facilitates teasing apart various modules. In our experiments, we only considered a small set of indicative experiments to show the utility of the proposed framework. Moreover, we only considered the activation blocks with less granularity, and a better localization may exist when performing path patching by analyzing the role of individual attention heads. Furthermore, we only used `phi-2` for the localization experiments, and more analysis would be required for other open-weight models that show a decent performance over the created commonsense queries. At last, we would like to mention that these experiments only provide a weak signal that localization may exist, and the current method of direct computation may not be transparent enough to find the decision-making modules for common sense reasoning. We encourage future works to consider finding the underlying circuits behind the commonsense reasoning. We believe the proposed framework will lead to a helpful resource with high utility, both for robust evaluation and circuit discovery of commonsense reasoning, helping find out ways in which these models can be made more accurate for commonsense reasoning in general.

## Ethical Aspects

Our work does not have any negative impact on the society. We create a dataset for evaluating LLMs for commonsense knowledge and evaluate open-weight LLMs exhaustively and rigorously.

## References


Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Aditya Chattopadhyay, Piyushi Manupriya, Anirban Sarkar, and Vineeth N. Balasubramanian. 2019. Neural network attributions: A causal perspective. In *International Conference on Machine Learning*.

Yejin Choi. 2022. The Curious Case of Commonsense Intelligence. *Daedalus*, 151(2):139–155.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

E. B. Fowlkes and C. L. Mallows. 1983. A method for comparing two hierarchical clusterings. *Journal of the American Statistical Association*, 78(383):553–569.

Lea Frermann, Ivan Titov, and Manfred Pinkal. 2014. A Hierarchical Bayesian Model for Unsupervised Induction of Script Knowledge. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*.

Kanishk Gandhi, Dorsa Sadigh, and Noah D. Goodman. 2023. Strategic reasoning with language models. *ArXiv*, abs/2305.19165.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. Localizing model behavior with path patching.

Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models.

Stefan Heimersheim and Neel Nanda. 2024. How to use and interpret activation patching.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs.

Bram Jans, Steven Bethard, Ivan Vulić, and Marie Francine Moens. 2012. Skip N-grams and Ranking Functions for Predicting Script Events. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.

Mojan Javaheripi, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, Suriya Gunasekar, Piero Kauffmann, Yin Tat Lee, Yuanzhi Li, Anh Nguyen, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Michael Santacroce, Harkirat Singh Behl, Adam Taumann Kalai, Xin Wang, Rachel Ward, Philipp Witte, Cyril Zhang, and Yi Zhang. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Abhinav Joshi, Areeb Ahmad, and Ashutosh Modi. 2024. COLD: Causal reasoning in closed daily activities. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Abhinav Joshi, Areeb Ahmad, Umang Pandey, and Ashutosh Modi. 2023. Scriptworld: Text based environment for learning procedural knowledge. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 5095–5103. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Xinze Li, Yixin Cao, Muhao Chen, and Aixin Sun. 2023. Take a break in the middle: Investigating subgoals towards hierarchical script generation.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark.

Aman Madaan, Dheeraj Rajagopal, Niket Tandon, Yiming Yang, and Eduard Hovy. 2021. Could you give me a hint ? generating inference graphs for defeasible reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5138–5147, Online. Association for Computational Linguistics.

Thomas McGrath, Matthew Rahtz, Janos Kramar, Vladimir Mikulik, and Shane Legg. 2023. The hydra effect: Emergent self-repair in language model computations.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2023. Locating and editing factual associations in gpt.

Ashutosh Modi. 2016. Event Embeddings for Semantic Script Modeling. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*.

Ashutosh Modi. 2017. *Modeling Common Sense Knowledge via Scripts*. Ph.D. thesis, Saarland University.

Ashutosh Modi, Tatjana Anikina, Simon Ostermann, and Manfred Pinkal. 2016. InScript: Narrative texts annotated with script information. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3485–3493, Portorož, Slovenia. European Language Resources Association (ELRA).

Ashutosh Modi and Ivan Titov. 2013. Learning semantic script knowledge with event embeddings. *arXiv preprint arXiv:1312.5198*.

Ashutosh Modi and Ivan Titov. 2014. Inducing Neural Models of Script Knowledge. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*.

Ashutosh Modi, Ivan Titov, Vera Demberg, Asad Sayeed, and Manfred Pinkal. 2017. Modeling Semantic Expectation: Using Script Knowledge for Referent Prediction. *Transactions of the Association for Computational Linguistics*.

Yasumasa Onoe, Michael J. Q. Zhang, Shankar Padmanabhan, Greg Durrett, and Eunsol Choi. 2023. Can lms learn new entities from descriptions? challenges in propagating injected knowledge.

Simon Ostermann, Ashutosh Modi, Michael Roth, Stefan Thater, and Manfred Pinkal. 2018. MCScript: A novel dataset for assessing machine comprehension using script knowledge. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Simon Ostermann, Sheng Zhang, Michael Roth, and Peter Clark. 2019. Commonsense inference in natural language processing (COIN) - shared task report. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 66–74, Hong Kong, China. Association for Computational Linguistics.

Jae Sung Park, Chandra Bhagavatula, Roozbeh Mottaghi, Ali Farhadi, and Yejin Choi. 2020. Visualcomet: Reasoning about the dynamic context of a still image. In *In Proceedings of the European Conference on Computer Vision (ECCV)*.

Judea Pearl. 2012. The do-calculus revisited. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, UAI'12, page 3–11, Arlington, Virginia, USA. AUAI Press.

Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. 2016. *Causal inference in statistics: A primer*. John Wiley & Sons.

Karl Pichotta and Raymond J. Mooney. 2016. Using Sentence-Level LSTM Language Models for Script Inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Gabriel Poesia, Kanishk Gandhi, E. Zelikman, and Noah D. Goodman. 2023. Certified reasoning with language models. *ArXiv*, abs/2306.04031.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5043–5053, Hong Kong, China. Association for Computational Linguistics.

Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. 2021. Timedial: Temporal commonsense reasoning in dialog.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

William M. Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A. Smith, and Yejin Choi. 2018. Event2Mind: Commonsense inference on events, intents, and reactions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 463–473, Melbourne, Australia. Association for Computational Linguistics.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning Script Knowledge with Web Experiments. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Joshua Robinson and David Wingate. 2023. Leveraging large language models for multiple choice question answering. In *The Eleventh International Conference on Learning Representations*.

Rachel Rudinger, Vera Demberg, Ashutosh Modi, Benjamin Van Durme, and Manfred Pinkal. 2015. Learning to predict script events from domain-specific text. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics*.

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021. ExplaGraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7716–7740, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Keisuke Sakaguchi, Chandra Bhagavatula, Ronan Le Bras, Niket Tandon, Peter Clark, and Yejin Choi. 2021. proscript: Partially ordered scripts generation via pre-trained language models.

Abhilasha Sancheti and Rachel Rudinger. 2022. What do Large Language Models Learn about Scripts? In *Proceedings of the 11th Joint Conference on Lexical and Computational Semantics*.

Roger C. Schank. 1975. The structure of episodes in memory. In DANIEL G. BOBROW and ALLAN COLLINS, editors, *Representation and Understanding*, pages 237–272. Morgan Kaufmann, San Diego.

Roger C. Schank and Robert P. Abelson. 1975. Scripts, Plans, and Knowledge. In *Proceedings of the 4th International Joint Conference on Artificial Intelligence*, IJCAI.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2021. CommonsenseQA 2.0: Exposing the limits of AI through gamification. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6076–6085, Hong Kong, China. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2009. Information theoretic measures for clusterings comparison: is a correction for chance necessary? In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 1073–1080, New York, NY, USA. Association for Computing Machinery.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small.

Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. 2024. Can llms reason with rules? logic scaffolding for stress-testing and improving llms.

Lilian D. A. Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. 2016. A Crowdsourced Database of Event Sequence Descriptions for the Acquisition of High-quality Script Knowledge. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*.

Peter West, Ronan Bras, Taylor Sorensen, Bill Lin, Liwei Jiang, Ximing Lu, Khyathi Chandu, Jack Hessel, Ashutosh Baheti, Chandra Bhagavatula, and Yejin Choi. 2023. NovaCOMET: Open commonsense foundation models with symbolic knowledge distillation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1127–1149, Singapore. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019b. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Rowan Zellers, Ximing Lu, Jack Hessel, Youngjae Yu, Jae Sung Park, Jize Cao, Ali Farhadi, and Yejin Choi. 2021. Merlot: Multimodal neural script knowledge models.

Wenting Zhao, Justin T Chiu, Jena D. Hwang, Faeze Brahman, Jack Hessel, Sanjiban Choudhury, Yejin Choi, Xiang Lorraine Li, and Alane Suhr. 2024. Uncommonsense reasoning: Abductive reasoning about uncommon situations.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning.

Wangchunshu Zhou, Ronan Le Bras, and Yejin Choi. 2023. Commonsense knowledge transfer for pretrained language models.

# Appendix

## Table of Contents


A Details to Graphical Representation . . . . 17

B Improving Data Quality. . . . . . . . . . . . . 17

C Prompt Templates . . . . . . . . . . . . . . . . . 17

D Trajectory Entropy . . . . . . . . . . . . . . . . 17

E Experimental Setup: Evaluating LLMs. . 18

F Hyperparameters for Fine-Tuning Experiments . . . . . . . . . . . . . . . . . . . . . . . . 19

G Additional Results and Empirical Findings . . . . . . . . . . . . . . . . . . . . . . . . . 19


## List of Figures


7 Compact Graph “baking a cake” 18

8 Input prompt formats for the MCQA-based evaluation. . . . . . 19

9 Formation of Conjugate prompt from a Clean prompt. . . . . . . . 19

10 Variation of trajectory entropy against the number of paths . . . . 20

11 Comparing the success rates of the models across task completion percentage. The error bands show +1 and -1 standard deviations across scenarios and in-context shots. . . 22

12 The figure shows the direct effect of path patching from the clean run to the random run (‘*going bowling*’). The peaks/deviations are less decisive than 5, highlighting the effectiveness of using the proposed conjugate prompts. . . . . 22

13 Success rate of the models after fine-tuning it on the MCQA dataset. The error bands show +1 and -1 standard deviation across scenarios. . . . . . . . . . . . . . 22

14 `Flying an airplane` and `Going grocery shopping` and show considerable improvement in `phi2-2.7b` and `Mistral-7b` when going from 0-shot to 5-shot. 22

15 Comparing the success rates of the models on all the scenarios based on their release date and model size. The size of each circle is indicative of the number of parameters in the model. Here, we observe that `phi2` shows a considerable gap in performance when compared to `Llama` model series and is very close to `Mistral-7b` despite having less than half the number of parameters. . . . . . . 22

16 Comparing success rates of the presented 6 models across each scenario and task completion percentages in a 5-shot setting. Here we see that for many scenarios `phi2-2.7b` and `Mistral-7b` show similar success rates. All the models have a high success rate earlier in each task, however as the task progresses the models show a drop in success. `gptj-6b` and `gptneo-1.3b` show almost random success ($\approx 50\%$) on each task. . . . . . . . . . . . . . . . . . . 23

17 Task completion % vs success rate of all models on each scenario averaged over all number of in-context examples, i.e. n-shots . . . . . . . 24

18 Task completion % vs success rate of all models on each scenario for fine-tuned `Llama3-8b` and `phi2-2.7b` . . . . . . . . . . 25

19 The figure shows the direct effect of path patching from the clean run to the conjugate run, leading to deviations starting at layer 20 and reinforced by the following layers (maximum deviations observed at layer 26 and layer 28), highlighting the role of particular layers in commonsense reasoning. . . . . . 26

## A Details to Graphical Representation

**Graphical Representation:** The created directed acyclic graphs (DAGs) provide a medium for generating enormous trajectories (scales from $1.6e + 16$ to $2.6e + 30$, also see Table 1), that are coming directly from human annotations (alignment annotation as well as the ESDs written by crowd-soured workers), providing us a proxy to represent the understanding of daily activities. Each node in the presented graph also contains miniature steps. For example, for the subtask "`take medicine`" (represented by a single node in the entire graph), some crowdsource workers explain it in more detail, like "`open water bottle,`" "`put medicine in the mouth,`" "`drink water.`" To handle such cases, we expand the node further, incorporating such substeps. This essentially leads to an enormous number of paths/ESDs from the start to the end node in the graph.

**Computing the total number of ESDs:** Note that the total number of ESDs that can be generated using the created graph is the total number of paths/trajectories from the start node to the end task node. (see Figure (7) for reference) The obtained DAGs can be used to compute the total number of paths using a simple DFS scheme. For considering the miniature steps as well, we expand the same graph by incorporating the parallel paths for all sub-tasks. Further, the total number of paths is computed considering every node in the compact graph that points to multiple text instances written by different human experts. We further use these paths to get multiple commonsense reasoning question prompts. Considering the humongous number of queries, we believe the generated examples act as a closed set that captures a proxy for the commonsense understanding related to a task.

## B Improving Data Quality

A noteworthy point about the created dataset is that although it is generated using an algorithmic procedure, the core knowledge still comes from humans. To cross-validate the quality of the generated dataset, we perform additional checks of the created DAGs by manual inspection of compact graph structure (and improve the quality by manually removing the nodes/entries/edges that do not form an explainable path from the start node to the end node), manual inspection of the descriptions that are clubbed together. Lastly, we conducted a sanity check, where we took a sample of 1k commonsense queries for 5 of 37 scenarios and asked 5 human annotators to know how well humans perform on the created task. We recorded an average accuracy of $95\%$ with $98\%$ being the maximum (more details in Table 2).

| Human Experts | Task Accuracy |
|---|---|
| Expert-1 | 98.00 |
| Expert-2 | 96.20 |
| Expert-3 | 96.52 |
| Expert-4 | 94.36 |
| Expert-5 | 92.00 |
| Average | 95.42 |
| claude-3-5-sonnet-20240620 | 94.30 |

Table 2: Performance of multiple annotators over the selected 1k samples (200 samples for 5 scenarios) over the generated commonsense queries for 5 activities. The high performance numbers indicate the presence of valid commonsense queries, well answerable by humans.

## C Prompt Templates

Fig. 8 shows the evaluation prompt template used for MCQA-based evaluation. The prompt is intentionally structured so that the LLM is intended to predict a single-choice token (Such as 'A,' 'B,' etc.).

## D Trajectory Entropy

To quantify the complexity across various scenarios and compare the created DAGs in detail, we define the trajectory entropy of a scenario. Trajectory entropy $\mathcal{H}_t$ for an DAG (Directed Acyclic Graph) $G$ is computed as:

$$\mathcal{H}_t = -\sum_{k=1}^{N} p(t_k) \log p(t_k)$$

Where $N$ is the total number of paths from the start to the end node $p(t_k)$, is the probability of trajectory $t_k$ defined as $p(t_k) = \prod_{ij} T(e_i \rightarrow e_j)$. $T(e_i \rightarrow e_j)$ is transition probability from event $e_i$ to $e_j$, which is defined uniformly across all the outgoing edges. Figure 10 shows the computations across multiple scenarios. We find that though there is a relationship between the entropy and the number of paths, there are a few outliers like '`playing tennis`', '`ironing laundry`', and '`renovating a room`', and the entropy would be another measure to identify the complexity of the task captured by the compact graph representations.

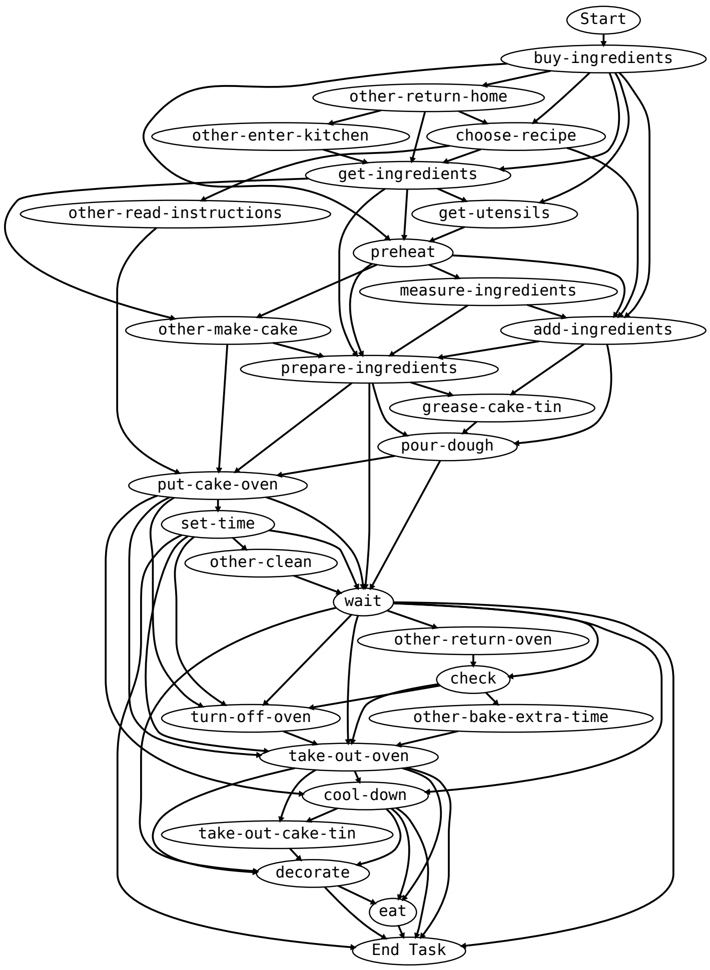


Figure 7: The figure shows the generated graph for the scenario "`baking a cake`".

## E Experimental Setup: Evaluating LLMs

As explained in the main paper, our overall evaluation relies on an MCQA-based Evaluation scheme that can generate $\sim 10^{17}$ commonsense queries for a single activity.

**Commonsense Queries for Evaluation:** Note that though the proposed scheme is capable of generating enormous queries, we perform all the analysis on the dataset generated from 2k trajectories (leading to $\sim 20k$ commonsense queries) for each scenario. We freeze this dataset of ($\sim 20k$ commonsense queries per scenario) for easier replicability of the obtained results.

We provide details of the additional experiments to investigate the effectiveness of these open-weight models below.

**In-Context Learning:** In recent years, LLMs have shown a surprising ability to capture the context via a few examples of the task provided in a prompt in the form of input-output examples (Dong et al., 2022). The LLMs predict the next output conditioning on the previous examples. To quantify the performance of the created task, it becomes important to consider evaluating LLMs using in-context examples. We perform an evaluation of the created commonsense queries by considering 1-shot, 2-shot, and 5-shot experiments for the LLMs. Previously, a few of the works (Brown et al., 2020; Robinson and Wingate, 2023) have reported significant boosts in performance when provided with in-context examples for MCQA-based evaluation.

**Fine-Tuning:** We also consider the finetuning of 2 open-weight models (`phi-2` and `Llama-3`) over a small set of 1000 queries from the created commonsense reasoning queries. We specifically choose 5 common scenarios and fine-tune the LLMs for an epoch. The fine-tuned scenarios include `planting a tree`, `going on a train`, `going grocery shopping`, `flying in an airplane`, and `riding on a bus`. We choose these five scenarios based on their generic nature, when compared to more

```
[ in-context examples (if few-shot/in-context learning experiment) ]
Question: For the task activity name, if the following steps are already completed in order
1. E_1, 2. E_2, 3. ... i. E_i, what should be the next suitable step for completing the task?
A. E_{i+1}
B. wrong choice sampled from the scenario
Answer: A
```

```
[ in-context examples (if few-shot/in-context learning experiment) ]
Question: For the task planting a tree, if the following steps are already completed in order
1. 'Go to garden center', 2. 'Obtain seedling.', what should be the next suitable step for
completing the task?
A. 'Water tree'
B. 'Find a location to plant tree'
Answer: B
```

Figure 8: Input prompt formats for the MCQA-based evaluation of autoregressive open-weight models (e.g., `llama(-2)`, `GPT-J`, etc.). The `black text` is the templated input. The `orange text` is the input from the current event trajectory, where the `activity name` denotes the description of the activity like `baking a cake`, or `planting a tree`. The next-token prediction probabilities of the option IDs at the `red text` is used as the observed prediction distribution.

```
Question: For the task activity name, if the following steps are already completed in order
1. E_1, 2. E_2, 3. ... p. E_p, what should be the next suitable step for completing the task?
A. E_{p+1}
B. E_{q+1}
Answer: A
```

```
Question: For the task activity name, if the following steps are already completed in order
1. E_1, 2. E_2, 3. ... q. E_q, what should be the next suitable step for completing the task?
A. E_{p+1}
B. E_{q+1}
Answer: B
```

Figure 9: Formation of Conjugate prompt from a Clean prompt. The `black text` is the template input ($x_\epsilon$), where the **`activity name`** denotes the description of the activity like `baking a cake`, or `planting a tree`. The `blue text` is the clean run ($x_{traj.}$) ending at step $E_p$, making $E_{p+1}$ to be the correct choice. The conjugate run input (`orange text`) is framed from a conjugate trajectory ($\bar{x}_{traj.}$) ending at $E_q$, making $E_{q+1}$ to be the correct conjugate choice. Note that in both prompts (clean and conjugate), the options ($x_{options}$) remain the same, i.e., $E_{p+1}$ and $E_{q+1}$ and only the clean trajectory is changed to conjugate trajectory. The next-token prediction probabilities of the option IDs at the `red text` is used as the observed prediction distribution. The change in the decision is monitored via the difference in logits corresponding to tokens ' A' and ' B' before and after the activation path patching.

specific scenarios like `'sewing a button'` or `'renoating a room'`.

**Generalization between Similar Scenarios:** To assess if simple finetuning over a few scenarios helps the model learn the MCQA evaluation format, we consider evaluating the fine-tuned models over all the available scenarios. This also helps validate if there is a generalization between similar scenarios, i.e., learning a scenario helps improve the performance over other similar scenarios.

## F Hyperparameters for Fine-Tuning Experiments

We employed the following hyperparameters to fine-tune our models. We set the batch size to 4 and utilized gradient accumulation steps of 4. The models were trained for one epoch with a learning rate of $1e-5$. A weight decay of 0.01 was applied. Flash attention (Dao et al., 2022) was enabled to enhance the training efficiency. The AdamW (Loshchilov and Hutter, 2017) optimizer was used for updating the model weights.

## G Additional Results and Empirical Findings

**Relation with Model Size:** Fig. 15 underscores the success rate of a model compared to its size and release date. We observe a surprising trend in that `phi2-2.7b` can outperform the Llama series of models despite its smaller size. Through Fig.

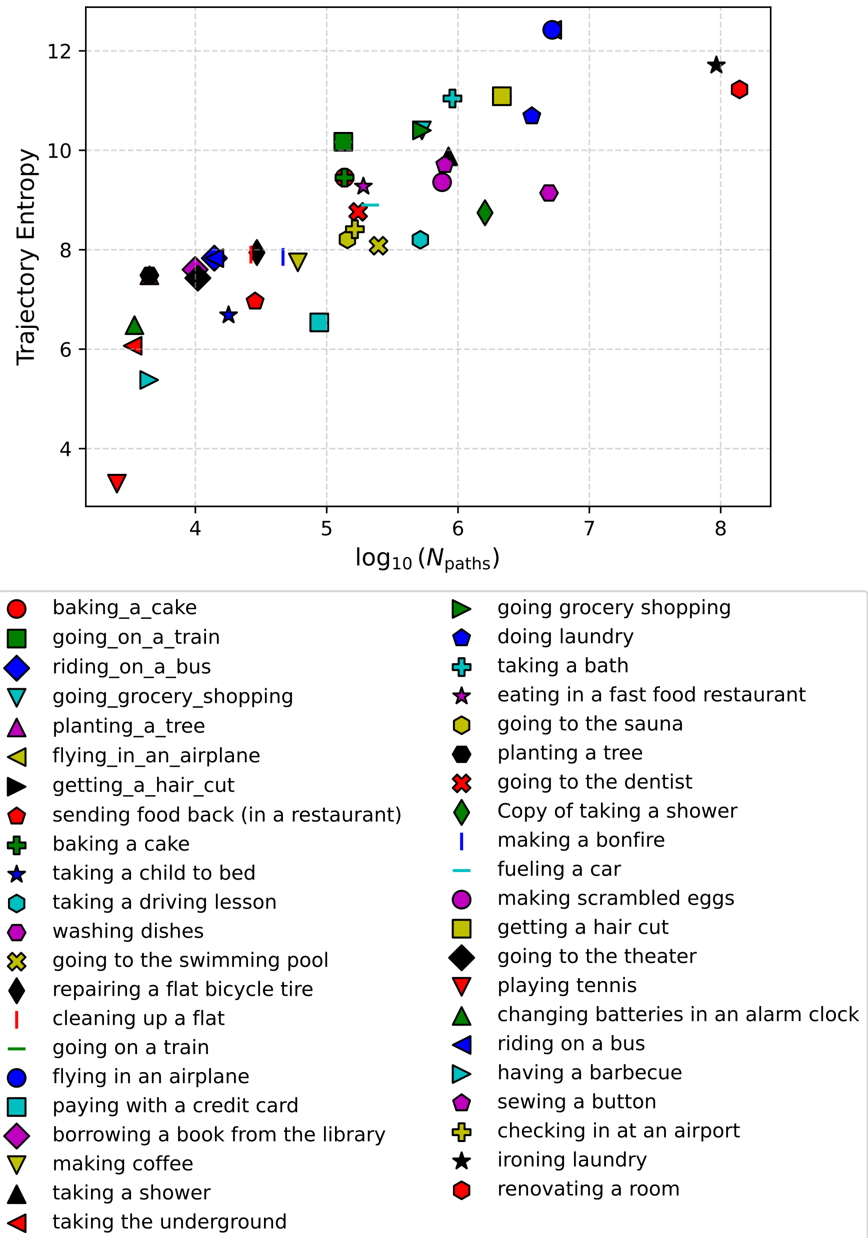


Figure 10: The figure shows the variation of trajectory entropy with the Total Number of Paths in the Compact Graph. We observe that some scenarios have less or almost equal trajectory entropy despite having a significantly higher number of paths. This demonstrates that the complexity of the task is not only due to the number of paths, but some other factors also play a role in determining complexity.

15, we observe that this performance rise could be attributed to the release dates of the models and the availability of pre-training datasets.

**Relation with Task completion percentage:** The MCQA formulation of the commonsense knowledge about the activities is framed using the steps/events involved in the activity, where a subpart of the trajectory (with length $m$) is considered by taking a split at a step $n \in \{1, m\}$ and using steps $e_1, e_2, \ldots e_{n-1}$ as a part of a commonsense reasoning question and $e_n$ as the correct choice for the question. The task completion percentage is calculated based on the current event step $(n)$ with respect to the total steps $(m)$ in the sampled trajectory. More task completion percentage means more context of a particular task, i.e. the query contains more number of steps.

We investigate model performance in Fig. 11 by comparing the success rates of the models against the task completion percentage. We observe that all models perform well for smaller task completion percentage; however, as the task progresses, all the models show a dip in success rates. This could possibly be attributed to either the long context of all the previous actions or the task's complexity as it progresses. In general, LLMs are expected to perform well with more context about the task (note the context length here does not increase by a significant margin). However, in this case, as the task progresses, the number of valid options increases with more variability, increasing the complexity of the commonsense queries.

To further explore how the performance varies with task completion percentage for different scenarios, we compare the performance across all the scenarios. Fig. 16 shows the success rate of each model across task completion percentages for each scenario. We observe a similar trend and notice that all models perform well initially but show a decline in performance thereafter.

We observed a few interesting trends when inspecting them across similar scenarios. For the scenarios that contain relationships with food, for example, in scenarios like `Making scrambled eggs`, `Baking a cake`, `Having a barbecue`, `Making coffee`, etc., the `Mistral-7b` shows a significant drop in the performance towards the end, highlighting the role of context in making the task more detailed and difficult to reason about. Moreover, we also find an interesting trend where the scenarios contain some movement, e.g., `Taking a driving lesson`, `Going to the theatre`, `Going bowling`, `Taking a child to bed`, `Going to the dentist`, `Riding on a bus`, `Flying in an airplane`, and `Checking in at the airport`; `Mistral-7b` and `phi2-2.7b` show improvements in success rates at the middle sections of the task, making the context more important for such scenarios.

| *Language Model* | 0-shot | 1-shot | 2-shot | 5-shot |
|---|---|---|---|---|
| gpt-j-6B | 50.19 | 50.14 | 50.59 | 50.05 |
| gpt-neo-1.3B | 50.07 | 49.58 | 49.86 | 50.26 |
| Llama-2-7b-chat-hf | 55.67 | 54.63 | 56.11 | 56.59 |
| Mistral-7B-v0.1 | 66.76 | 67.61 | 70.24 | 71.13 |
| **Average** | **55.99** | **55.78** | **57.04** | **57.26** |

Table 3: Average performance for In-context learning experiments over multiple open-weight models.

**Improvements with In-Context Learning Examples:** Fig. 14 shows the improvement of the models from zero-shot to five-shot settings, especially at the initial steps. `Mistral-7b`, `phi2-2.7b`, `Llama3-8b`, and `gptj-6b` show performance improvements in the `Going Grocery Shopping` scenario (holding the highest scores in `Mistral-7b`). However, `Llama2-7b` shows performance degradation when going from 0-shot to 5-shot . A similar trend is observed in `Flying in an Airplane` scenario, with `Mistral-7b` and `phi2-2.7b` showing performance improvements while Llama models show a degradation in performance .

**Improvements with Finetuning:** Table 5 and Fig. 13 highlights the improvement of `Llama3` across all scenarios and task completion status upon fine-tuning. We observe that after fine-tuning `Llama3`, it has a rise in success rate across time steps and also outperform `Mistral-7B` when prompted with in-context examples. Fig. 18 dives deeper into the fine-tuned models across time steps for each scenario. We also observe the same rising trend, suggesting that the model generalize upon fine-tuning. However, we observe a decrease in performance of `phi2-2.7b` in general and across time steps. We observe the same trend in Fig. 18 for `phi2-2.7b` in all scenarios.

**Generalization across Multiple Scenarios:** We fine-tuned `Llama3-8b` and `phi2-2.7b` on the trajectories from `Going grocery shopping` and evaluated the models on all the scenarios. Fig. 13 highlights that `Llama3-8b` generalizes to all the scenarios, especially across time steps. However, we see a drop in the performance of `phi2-2.7b` in general and across time steps, pointing towards low generalization capability of smaller models.

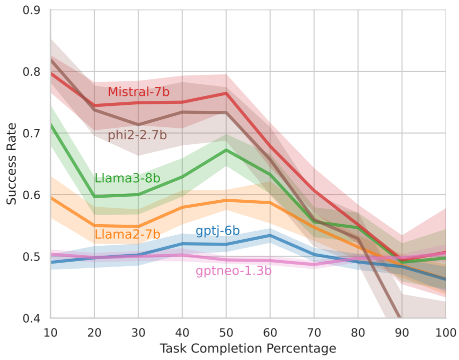


Figure 11: Comparing the success rates of the models across task completion percentage. The error bands show +1 and -1 standard deviations across scenarios and in-context shots.

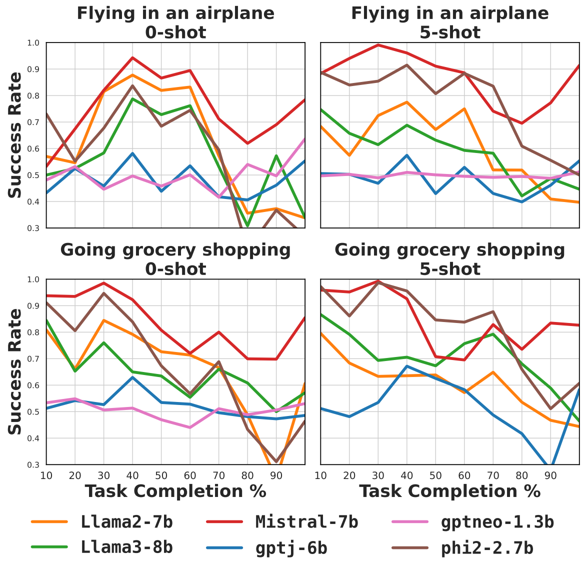


Figure 14: `Flying an airplane` and `Going grocery shopping` and show considerable improvement in `phi2-2.7b` and `Mistral-7b` when going from 0-shot to 5-shot.

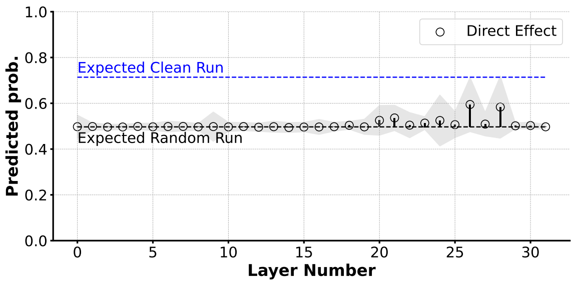


Figure 12: The figure shows the direct effect of path patching from the clean run to the random run (‘*going bowling*’). The peaks/deviations are less decisive than 5, highlighting the effectiveness of using the proposed conjugate prompts.

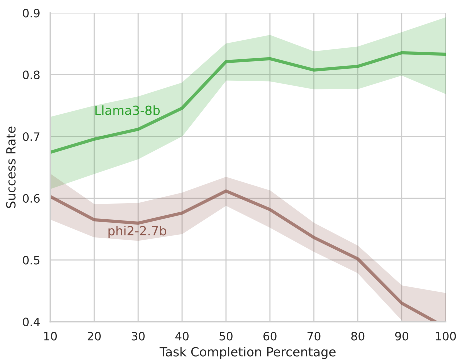


Figure 13: Success rate of the models after fine-tuning it on the MCQA dataset. The error bands show +1 and -1 standard deviation across scenarios.

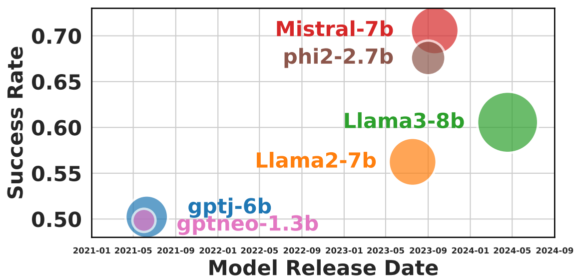


Figure 15: Comparing the success rates of the models on all the scenarios based on their release date and model size. The size of each circle is indicative of the number of parameters in the model. Here, we observe that `phi2` shows a considerable gap in performance when compared to `Llama` model series and is very close to `Mistral-7b` despite having less than half the number of parameters.

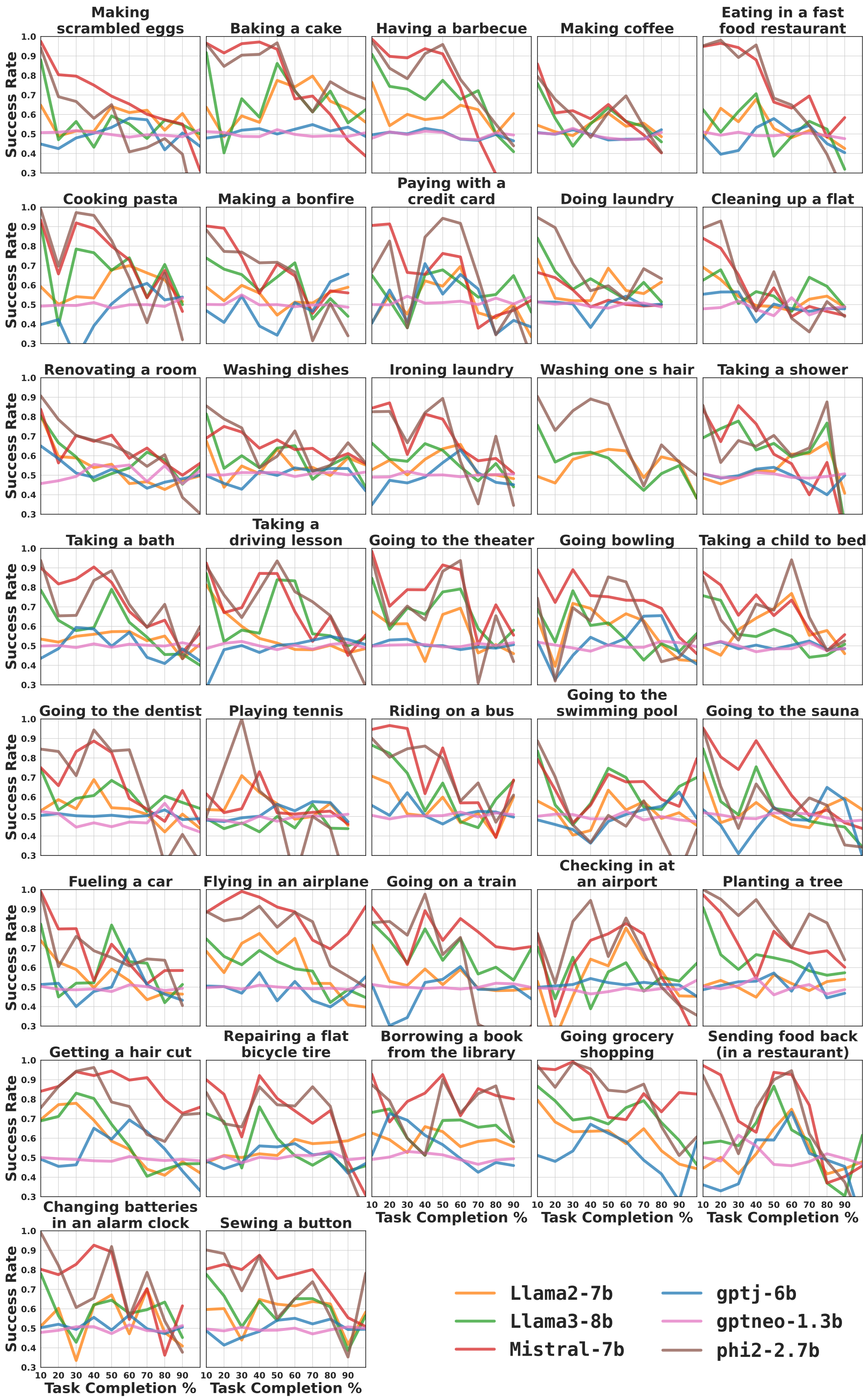


Figure 16: Comparing success rates of the presented 6 models across each scenario and task completion percentages in a 5-shot setting. Here we see that for many scenarios `phi2-2.7b` and `Mistral-7b` show similar success rates. All the models have a high success rate earlier in each task, however as the task progresses the models show a drop in success. `gptj-6b` and `gptneo-1.3b` show almost random success ($\approx 50\%$) on each task.

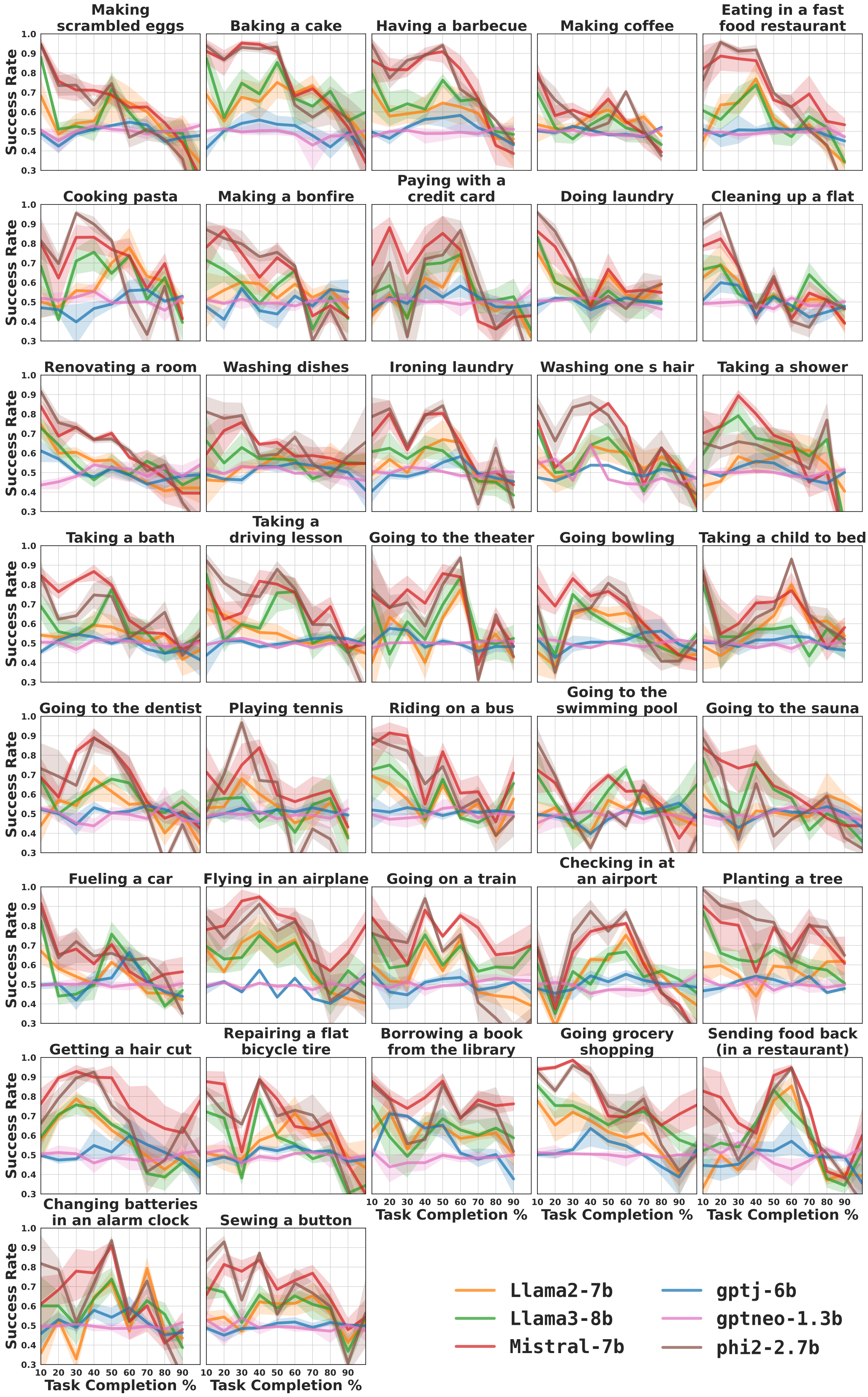


Figure 17: Task completion % vs success rate of all models on each scenario averaged over all number of in-context examples, i.e. n-shots

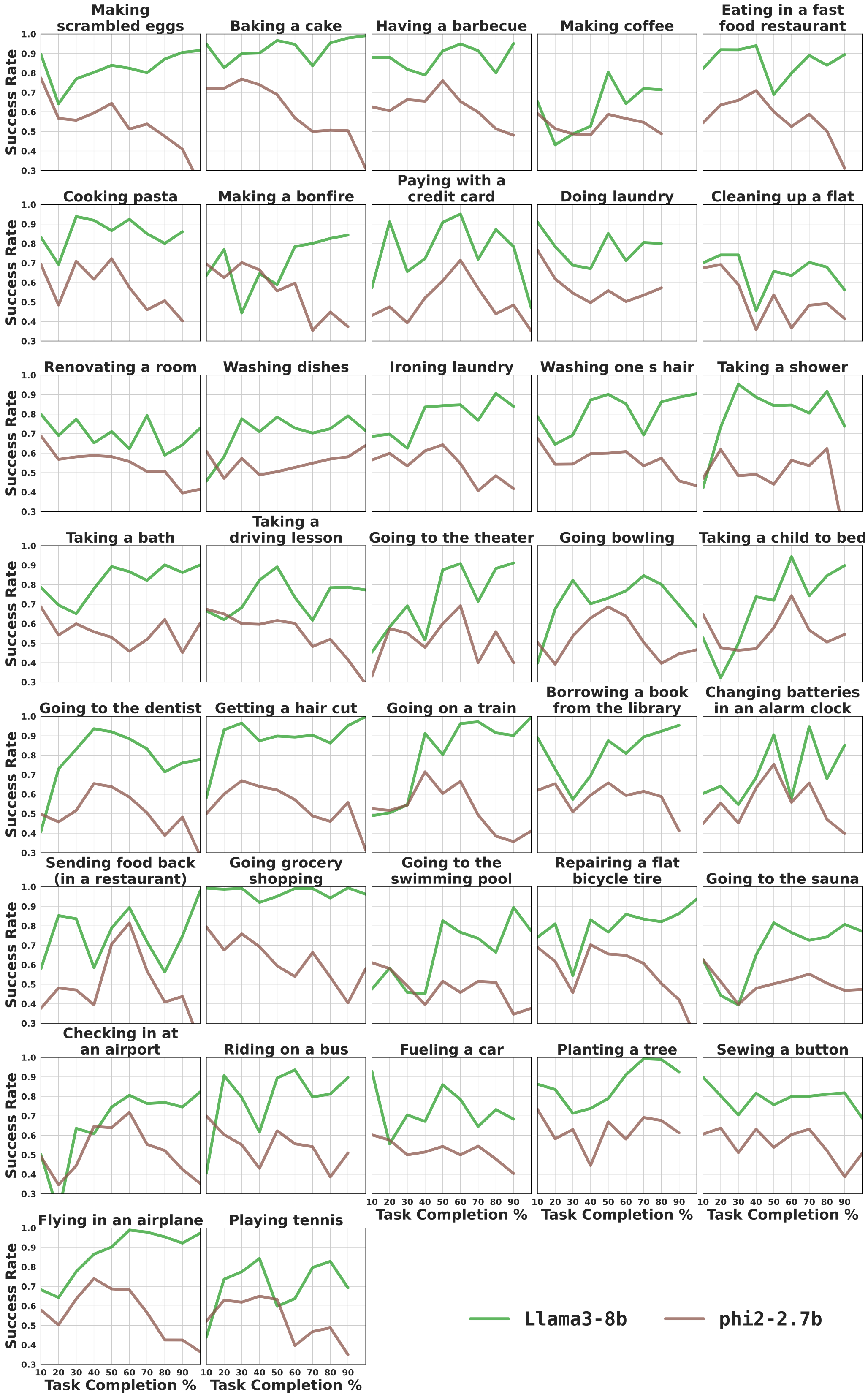


Figure 18: Task completion % vs success rate of all models on each scenario for fine-tuned `Llama3-8b` and `phi2-2.7b`

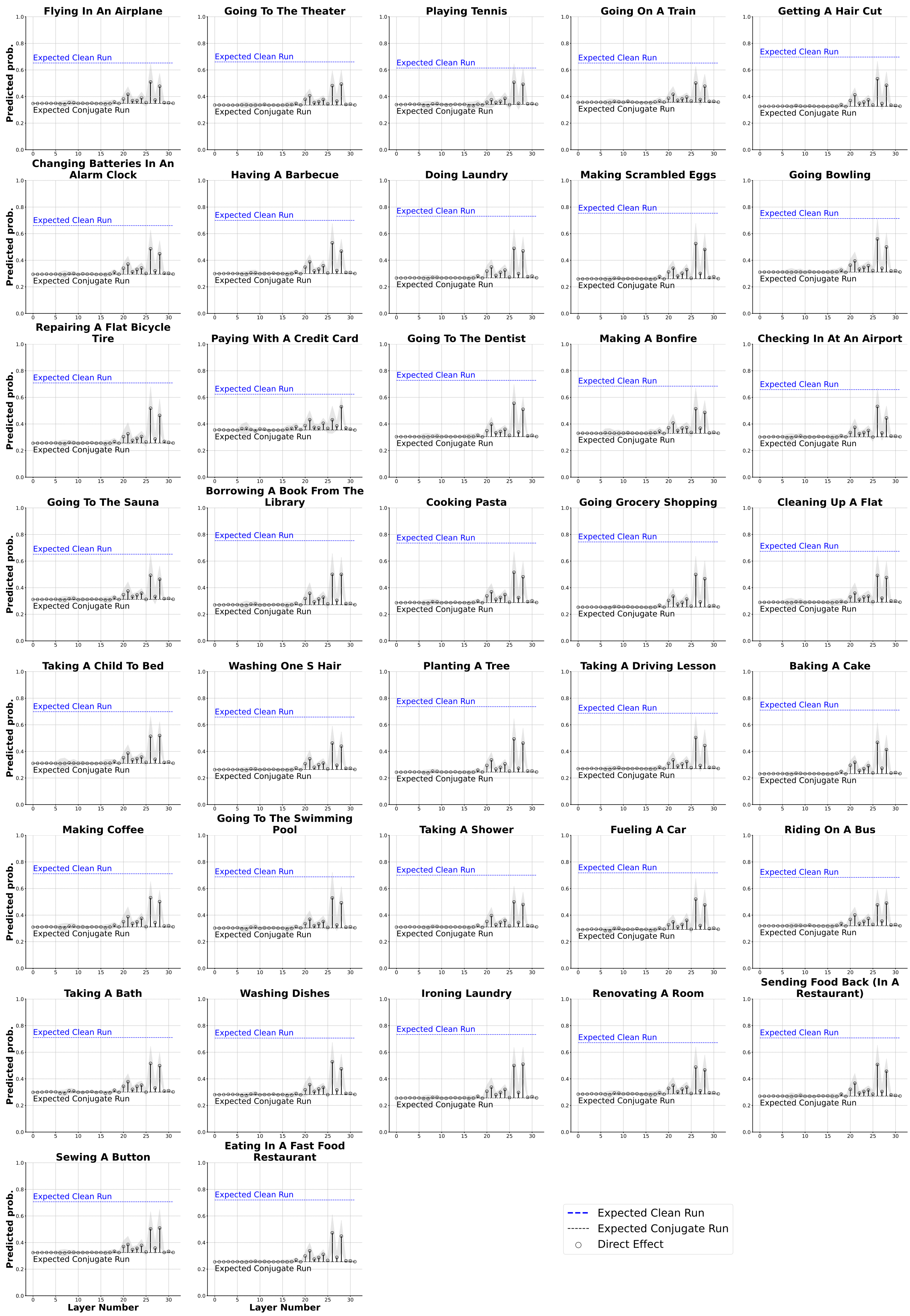


Figure 19: The figure shows the direct effect of path patching from the clean run to the conjugate run, leading to deviations starting at layer 20 and reinforced by the following layers (maximum deviations observed at layer 26 and layer 28), highlighting the role of particular layers in commonsense reasoning.

| Activity | gpt-neo-1.3B | phi-2 | gpt-j-6B | Llama-2-7b | Mistral-7B-v0.1 | Llama-3 |
|---|---|---|---|---|---|---|
| baking a cake | 48.86 | 73.78 | 44.43 | 69.28 | **77.84** | 63.11 |
| borrowing a book from the library | 49.37 | 60.76 | 52.45 | 61.86 | **75.08** | 55.41 |
| changing batteries in an alarm clock | 50.02 | **62.79** | 50.65 | 51.51 | 60.53 | 51.19 |
| checking in at an airport | 48.77 | 55.65 | 48.60 | 53.74 | **58.16** | 48.76 |
| cleaning up a flat | 49.23 | 57.93 | 51.29 | 50.64 | **59.58** | 51.84 |
| cooking pasta | 51.88 | 67.03 | 49.68 | 60.56 | **70.20** | 60.31 |
| doing laundry | 49.43 | 68.54 | 50.27 | 60.84 | **73.26** | 56.46 |
| eating in a fast food restaurant | 49.91 | 63.00 | 52.19 | 50.62 | **71.74** | 53.64 |
| flying in an airplane | 49.83 | 59.43 | 48.30 | 62.40 | **74.14** | 57.93 |
| fueling a car | 51.05 | 59.78 | 51.30 | 45.54 | **64.44** | 50.79 |
| getting a hair cut | 49.91 | 56.81 | 48.53 | 54.72 | **73.75** | 50.58 |
| going bowling | 49.87 | 58.05 | 50.59 | 53.14 | **61.15** | 49.29 |
| going grocery shopping | 50.84 | 70.96 | 53.12 | 67.36 | **84.96** | 66.77 |
| going on a train | 50.16 | 55.00 | 52.03 | 58.10 | **69.81** | 54.70 |
| going to the dentist | 50.29 | 54.28 | 51.56 | 52.10 | **66.58** | 52.57 |
| going to the sauna | 50.47 | 53.17 | 49.91 | 50.70 | **57.50** | 50.83 |
| going to the swimming pool | 47.90 | 54.34 | 49.51 | 46.93 | **57.16** | 46.72 |
| going to the theater | 48.49 | 52.76 | 51.94 | 48.54 | **61.35** | 47.75 |
| having a barbecue | 48.41 | **77.19** | 52.33 | 60.31 | 76.30 | 57.97 |
| ironing laundry | 51.69 | 61.57 | 48.65 | 57.88 | **65.04** | 51.81 |
| making a bonfire | 51.17 | **65.22** | 49.22 | 51.33 | 64.29 | 48.00 |
| making coffee | 51.62 | 57.51 | 49.44 | 51.04 | **59.95** | 49.91 |
| making scrambled eggs | 51.49 | **66.08** | 48.64 | 56.46 | 65.99 | 57.84 |
| paying with a credit card | 49.15 | 38.92 | 50.49 | 49.07 | **50.95** | 49.84 |
| planting a tree | 49.60 | 71.25 | 49.18 | 63.59 | **73.19** | 60.27 |
| playing tennis | 48.65 | 56.09 | 50.87 | 47.67 | **64.96** | 50.18 |
| renovating a room | 47.09 | 60.92 | 51.38 | 52.45 | **63.49** | 51.06 |
| repairing a flat bicycle tire | 50.38 | **71.32** | 50.26 | 59.05 | 69.59 | 55.59 |
| riding on a bus | 48.04 | 61.99 | 53.31 | 58.39 | **71.99** | 54.98 |
| sending food back (in a restaurant) | 53.98 | 49.23 | 48.37 | 50.84 | **63.69** | 51.15 |
| sewing a button | 51.95 | 63.06 | 48.53 | 54.28 | **66.68** | 52.70 |
| taking a bath | 49.91 | 59.31 | 49.54 | 55.32 | **69.52** | 52.74 |
| taking a child to bed | 51.56 | 60.21 | 49.55 | 54.69 | **68.74** | 54.25 |
| taking a driving lesson | 49.67 | **63.97** | 51.32 | 59.64 | 63.27 | 54.28 |
| taking a shower | 49.94 | 49.93 | 50.17 | 57.77 | **68.33** | 55.61 |
| washing dishes | 49.99 | **62.32** | 50.73 | 52.32 | 60.07 | 50.26 |
| washing one s hair | 48.63 | 64.52 | 50.15 | 53.40 | **66.75** | 57.12 |
| **Average Performance** | 50.01 | 60.67 | 50.23 | 55.27 | **66.76** | 53.63 |

Table 4: Success Rate (%) of various open-weight LLMs over the created commonsense queries for 37 real-world activities. Overall, we find Mistral-7B-v0.1 performing best over the maximum number of scenarios, highlighting better commonsense reasoning abilities when compared to other open-weight models. We also observe that phi-2, with a surprisingly lower number of parameters, outperforms models with more number of parameters.

| Scenario | planting a tree | going on a train | going grocery shopping | flying in an airplane | riding on a bus |
|---|---|---|---|---|---|
| baking a cake | 91.58 (↑ 28.47%) | 94.09 (↑ 30.98%) | 92.59 (↑ 29.48%) | 93.21 (↑ 30.10%) | 82.23 (↑ 19.12%) |
| borrowing a book from the library | 82.26 (↑ 26.85%) | 86.55 (↑ 31.14%) | 86.44 (↑ 31.03%) | 84.49 (↑ 29.08%) | 81.14 (↑ 25.73%) |
| changing batteries in an alarm clock | 81.17 (↑ 29.98%) | 79.09 (↑ 27.90%) | 74.13 (↑ 22.94%) | 74.86 (↑ 23.67%) | 75.23 (↑ 24.04%) |
| checking in at an airport | 62.29 (↑ 13.53%) | 67.31 (↑ 18.55%) | 67.07 (↑ 18.31%) | 68.58 (↑ 19.82%) | 61.22 (↑ 12.46%) |
| cleaning up a flat | 63.55 (↑ 11.71%) | 63.36 (↑ 11.52%) | 65.41 (↑ 13.57%) | 65.23 (↑ 13.39%) | 65.38 (↑ 13.54%) |
| cooking pasta | 84.45 (↑ 24.14%) | 83.15 (↑ 22.84%) | 86.65 (↑ 26.34%) | 83.62 (↑ 23.31%) | 82.37 (↑ 22.06%) |
| doing laundry | 76.06 (↑ 19.60%) | 80.14 (↑ 23.68%) | 79.08 (↑ 22.62%) | 79.29 (↑ 22.83%) | 76.09 (↑ 19.63%) |
| eating in a fast food restaurant | 78.84 (↑ 25.19%) | 84.95 (↑ 31.30%) | 85.69 (↑ 32.04%) | 87.35 (↑ 33.70%) | 80.45 (↑ 26.80%) |
| flying in an airplane | 86.81 (↑ 28.88%) | 90.15 (↑ 32.22%) | 84.44 (↑ 26.51%) | 95.71 (↑ 37.78%) | 77.50 (↑ 19.57%) |
| fueling a car | 73.19 (↑ 22.40%) | 73.07 (↑ 22.28%) | 73.44 (↑ 22.65%) | 74.48 (↑ 23.69%) | 73.57 (↑ 22.78%) |
| getting a hair cut | 84.83 (↑ 34.25%) | 85.45 (↑ 34.87%) | 85.60 (↑ 35.02%) | 87.89 (↑ 37.31%) | 79.17 (↑ 28.59%) |
| going bowling | 66.00 (↑ 16.71%) | 69.63 (↑ 20.34%) | 67.37 (↑ 18.08%) | 69.89 (↑ 20.60%) | 68.66 (↑ 19.37%) |
| going grocery shopping | 90.69 (↑ 23.92%) | 92.69 (↑ 25.92%) | 96.82 (↑ 30.05%) | 94.17 (↑ 27.40%) | 87.66 (↑ 20.89%) |
| going on a train | 79.66 (↑ 24.96%) | 93.89 (↑ 39.19%) | 78.45 (↑ 23.75%) | 86.35 (↑ 31.65%) | 75.09 (↑ 20.39%) |
| going to the dentist | 68.95 (↑ 16.38%) | 78.18 (↑ 25.61%) | 77.02 (↑ 24.45%) | 79.14 (↑ 26.57%) | 71.23 (↑ 18.66%) |
| going to the sauna | 66.59 (↑ 15.76%) | 72.87 (↑ 22.04%) | 66.80 (↑ 15.97%) | 70.92 (↑ 20.09%) | 65.05 (↑ 14.22%) |
| going to the swimming pool | 66.51 (↑ 19.78%) | 69.39 (↑ 22.66%) | 63.75 (↑ 17.02%) | 71.40 (↑ 24.67%) | 62.85 (↑ 16.12%) |
| going to the theater | 71.62 (↑ 23.90%) | 74.62 (↑ 26.90%) | 71.81 (↑ 24.09%) | 80.23 (↑ 32.51%) | 68.64 (↑ 20.92%) |
| having a barbecue | 87.32 (↑ 29.34%) | 86.63 (↑ 28.65%) | 87.92 (↑ 29.94%) | 87.47 (↑ 29.49%) | 80.32 (↑ 22.34%) |
| ironing laundry | 73.86 (↑ 22.05%) | 78.18 (↑ 26.37%) | 78.57 (↑ 26.76%) | 78.68 (↑ 26.87%) | 77.30 (↑ 25.49%) |
| making a bonfire | 77.72 (↑ 29.72%) | 77.87 (↑ 29.87%) | 72.06 (↑ 24.06%) | 75.50 (↑ 27.50%) | 67.50 (↑ 19.50%) |
| making coffee | 70.11 (↑ 20.20%) | 64.84 (↑ 14.93%) | 63.53 (↑ 13.62%) | 61.54 (↑ 11.63%) | 58.30 (↑ 8.39%) |
| making scrambled eggs | 71.61 (↑ 13.77%) | 79.90 (↑ 22.06%) | 82.66 (↑ 24.82%) | 80.72 (↑ 22.88%) | 78.63 (↑ 20.79%) |
| paying with a credit card | 74.52 (↑ 24.68%) | 75.53 (↑ 25.69%) | 73.12 (↑ 23.28%) | 77.32 (↑ 27.48%) | 65.85 (↑ 16.01%) |
| planting a tree | 95.46 (↑ 35.18%) | 89.98 (↑ 29.70%) | 85.47 (↑ 25.19%) | 85.98 (↑ 25.70%) | 76.00 (↑ 15.72%) |
| playing tennis | 59.51 (↑ 9.33%) | 60.14 (↑ 9.96%) | 62.43 (↑ 12.25%) | 63.65 (↑ 13.47%) | 60.95 (↑ 10.77%) |
| renovating a room | 72.92 (↑ 21.86%) | 75.30 (↑ 24.24%) | 72.11 (↑ 21.05%) | 72.05 (↑ 20.99%) | 73.54 (↑ 22.48%) |
| repairing a flat bicycle tire | 80.63 (↑ 25.04%) | 83.22 (↑ 27.63%) | 80.89 (↑ 25.30%) | 82.12 (↑ 26.53%) | 77.30 (↑ 21.71%) |
| riding on a bus | 84.73 (↑ 29.75%) | 80.08 (↑ 25.10%) | 76.45 (↑ 21.47%) | 86.34 (↑ 31.36%) | 90.70 (↑ 35.72%) |
| sending food back (in a restaurant) | 70.47 (↑ 19.32%) | 73.10 (↑ 21.95%) | 71.93 (↑ 20.78%) | 64.22 (↑ 13.07%) | 66.61 (↑ 15.46%) |
| sewing a button | 76.73 (↑ 24.03%) | 81.05 (↑ 28.35%) | 80.59 (↑ 27.89%) | 77.56 (↑ 24.86%) | 76.23 (↑ 23.53%) |
| taking a bath | 77.71 (↑ 24.96%) | 85.67 (↑ 32.92%) | 81.47 (↑ 28.72%) | 81.65 (↑ 28.90%) | 77.57 (↑ 24.82%) |
| taking a child to bed | 68.52 (↑ 14.26%) | 74.16 (↑ 19.90%) | 69.31 (↑ 15.05%) | 70.35 (↑ 16.09%) | 68.11 (↑ 13.85%) |
| taking a driving lesson | 72.96 (↑ 18.67%) | 74.08 (↑ 19.79%) | 72.93 (↑ 18.64%) | 76.74 (↑ 22.45%) | 75.44 (↑ 21.15%) |
| taking a shower | 78.23 (↑ 22.62%) | 79.28 (↑ 23.67%) | 79.36 (↑ 23.75%) | 78.61 (↑ 23.00%) | 75.06 (↑ 19.45%) |
| washing dishes | 65.23 (↑ 14.99%) | 69.43 (↑ 19.19%) | 67.94 (↑ 17.70%) | 66.78 (↑ 16.54%) | 68.03 (↑ 17.79%) |
| washing one s hair | 76.49 (↑ 19.37%) | 84.70 (↑ 27.58%) | 82.76 (↑ 25.64%) | 78.70 (↑ 21.58%) | 79.09 (↑ 21.97%) |

Table 5: The table shows the performance of `Llama-3-8B` finetuned over 5 scenarios (presented in the columns). We observe a boost in performance (highlighted in blue) when compared to the MCQA-based evaluation.